Master slave concept in Jenkins:

1. Jenkins server and slave server must have java same version installed
2. In Jenkins server create another node

Manage Jenkins-manage node-new node-give some name-description

Root directory (some path like /opt/build)

Enter label (some name like agent-node)

1. Check
2. Save the node and click on node
3. Download the agent.jar file in system
4. Mush check the Jenkins server public ip in agent key public ip is same if not go into the configure system check the public ip
5. In mobaxterm download the agent.jar file in /home/ec2-user directory
6. Move the agent.jar to /opt directory

Mv /home/ec2-user/agent.jar /opt

1. Go into the /opt check the agent.jar file moving or not
2. copy the agent node path in Jenkins server paste the path in the slave machine /opt directory
3. in Jenkins server refresh the agent node
4. then create the job within the node the run the jobs
5. in slave node /opt/build path check the jobs are reflecting or not

**Jenkins Master and Slave configuration**

**🧰 Prerequisites**

1. Jenkins server
2. Slave server with Java installation

**Procedure:**

1. Goto Manage Nodes
   * Manage Jenkins --> Manage Nodes and Clouds --> New Node
2. Add the node name as Permanent Agent
3. Provide below information to add Jenkins agent  
   **Name:** uniquely identifies an agent within this Jenkins installation  
   **Description:**  
   **Number of executors:** 2  
   **Remote root directory:** /home/ec2-user/maven-agent  
   **Labels:** Labels (or tags) are used to group multiple agents into one logical group.  
   **Usage:**
   * Use this node as much as possible
   * Only build jobs with label expressions matching this node

**Launch method:**

* + Launch agent by connecting it to the master
  + Launch agent via execution of command on the controller

**Custom WorkDir path:** custom Remoting work directory will be used instead of the Agent Root Directory  
**- Use WebSocket [x]**  
**Availability:**

* + Keep this agent online as much as possible
  + Bring this agent online according to a schedule
  + Bring this agent online when in demand, and take offline when idle

1. Once you save above configuration you will get a command which should be executed in the agent. it contains agent.jar, a secret-file, and a jnlp file
2. echo "secret\_key" > secret-file

java -jar agent.jar -jnlpUrl http://<Jenkins\_URL>/computer/abc/jenkins-agent.jnlp -secret @secret-file -workDir "/home/ec2-user"

1. Once connected you can create or edit a job to chose this option in the Restrict where this project can be run

**Lessons Learned:**

1. Master and slave should have
   * same Java version
   * Same Maven version
2. In case Jave or Maven paths are referring to the agent system then aline it according to masters Global Tool Configuration
3. In case the case of the AWS server make sure your Jenkins URL should be updated to the latest Public IP.